Vol.2020-DPS-182 No.21
Vol.2020-CSEC-88 No.21
2020/3/12

深層学習モデルを用いた コマンドログに基づくユーザなりすまし検知

安達　貴洋[1]　小澤 誠一[1]　春木 博行[2]

概要：近年，不正アクセスによるサイバー攻撃が深刻化しており，その手口は正規のユーザになりすまして不正侵入することが多いことから，内部ユーザの異常行動を検知する仕組みが重要となっている．これに対し，事前にユーザに関する情報をプロファイリングし，実際に観測されたユーザの情報と照合することで，なりすましによる不正侵入を検知する手法が広く使われている．しかし，高い検知精度と低い誤警報率を実現し，攻撃者を想定した実践的なモデルを得ることは、依然として困難な課題である．本研究では，UNIX 系サーバーで収集された社員の実行コマンド・ログ情報に基づいてなりすまし判定するため，Character-level CNN に自己注意機構と BiLSTM を導入したモデルを構築し、その性能評価を行った。

キーワード：不正アクセス，なりすまし，深層学習，UNIX コマンド

Masquerade Detection Based on Users' Command Logs Using Deep Learning Models

Abstract: In recent years, cyberattacks by unauthorized access have become more serious, and attackers try to pretend to be normal users. Therefore, it is important to detect abnormal users by profiling user information. However, constructing a practical system with high detection accuracy and low false alarm rate is still challenging issues. In this work, we carry out an empirical study to evaluate the effectiveness of the following deep learning models: Character-level CNN, self attention mechanism, and BiLSTM. In the experiment, we conduct the performance evaluation for such deep learning models against the data set of comm and log collected from UNIX servers.

Keywords: Masquerade Detection, Deep Learning, UNIX command

1. まえがき

近年，不正アクセスによるサイバー攻撃が深刻化しており，異常なユーザを検知する仕組みの導入が求められている．総務省が公開している「不正アクセス行為の発生状況及びアクセス制御機能に関する技術の研究開発の状況」[1]によると，平成 27 年から不正アクセス違反は増加傾向にあり，検挙件数は平成 27 年の 373 件から平成 29 年は 648 件，検挙人員は平成 25 年の 147 人から平成 29 年は 255 人に増加したと報告されている．

不正アクセスの手口として，ブラウザや Web サイト，OS などの脆弱性を狙って情報を入手する手口も存在するが，最も多い手口は個人になりすまして不正に侵入する方法である．このなりすまし攻撃は，管理や設定の甘いパスワードを盗みだしたり，流出したアカウント情報を使用したりすることでシステムやサービスにログインする．そのため，不正アクセスの被害にあわない対策として，内部のネットワークやシステムなどに対して定期的な安全診断や監視をすることで異常なユーザを検知することが重要である．なりすまし攻撃による不正アクセスを検知するためには，事前にユーザに関する情報をプロファイリングし，実際に観測されたユーザの情報がプロファイル情報と照合して適切であるか判断する必要がある．

このプロファイル情報としては主にログイン時間やログイン場所，セッションの時間，入力されたコマンド等に注目されて，なりすまし攻撃を検知する研究は 10 年以上にわ

[1] 神戸大学大学院工学研究科
[2] LINE 株式会社

Vol.2020-DPS-182 No.21
Vol.2020-CSEC-88 No.21
2020/3/12

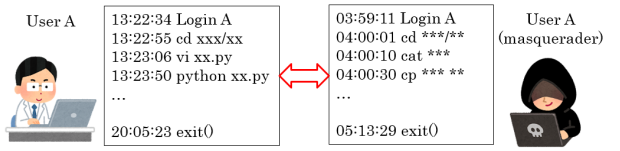

図 1　コマンドログに基づくなりすましユーザの検知

たって行われている．しかし，高い検知精度と低い誤警報率を実現すること，そして攻撃者を想定した実践的なモデルを作成し評価することは依然として困難な課題である．

なりすまし攻撃の検知において，コマンド情報はユーザの意図が反映された情報であるため，Kim ら [2] の Support Vector Machine を用いた研究や，Mahajan ら [3] による Hidden Markov Model を用いた研究など，UNIX コマンドを基にしつつ機械学習を活用した異常検知手法は従来より提案されている．また近年では，Elmasry ら [4] による Particle Swarm Optimization と深層学習モデルを活用した研究があるが，深層学習モデルを活用したなりすまし攻撃に関する研究は多くは存在しない．

そこで本研究では，UNIX コマンドを基に深層学習を活用してなりすまし攻撃を検知可能にするモデルを提案する．Elmasry らの実験結果を参考に，文字単位のベクトルを生成し畳み込みニューラルネットワークを用いて学習を行う Character-level CNN[5] に次の 3 点の変更を加えた，なりすまし検知モデルを提案する．(1) コマンドの引数との関係を学習するため，コマンド数，1 コマンドあたりの文字数，文字埋め込みベクトルの 3 種類の入力に対して畳み込む．(2) コマンドの類似性と共起性を学習するため，自己注意機構を組み込む．(3) 同じ入力コマンドでもそのシステム応答の違いからユーザの意図が反映されるため，入力コマンドに対するシステム応答情報を入力特徴とする．

2. 深層学習を用いたなりすまし検知モデル

2.1 なりすまし検知のフロー

なりすまし検知に用いるコマンドログデータの生成方法とその検知までの流れを説明する．本研究の目標は，図 1 のように，ユーザそれぞれに対して普段の入力コマンドを識別するモデルを作成することで，正規ユーザになりすまして不正アクセスした攻撃者を検知することである．なりすましユーザ検知の最終的な目標は，なりすましユーザを早期に検知し，攻撃が完了する前に防ぐことが理想であり，可能な限り短い間隔で検知し続ける必要がある．本研究では，コマンドログがストリーミング形式で収集され続ける環境を想定し，Alg.1 の流れでなりすましを検知するシステムを提案する．

この検知システムは，コマンドログを受け取ったとき過去の検知履歴にないコマンドが一定数 P に達したときに検知を開始する (Alg.1,1 行目)．攻撃者側が何コマンド以上入力をすれば攻撃を完了するのか未知であるが，仮に攻

Algorithm 1 なりすまし検知

Require: 対象ユーザのコマンドログ，検知履歴，未検知コマンド許容数 P, データ作成時間範囲 T，データ作成最大コマンド数 mx, データ作成最小コマンド数 mn, 閾値 σ, 前回の出力値 $Output_{t-1}$
Ensure: 入力コマンドとその入力時刻, $P \leq mx$

1: **if** 取得したコマンドログに検知履歴にない新たなコマンドが P 以上存在 **then**
2:　$Input_t \Leftarrow$ 最新の入力コマンドから T 以内に入力されたコマンド系列
3:　**if** $|Input_t| < mn$ **then**
4:　　break
5:　**end if**
6:　**if** $|Input_t| > mx$ **then**
7:　　$Input_t \Leftarrow Input$ で新しいものから順に mx 個
8:　**end if**
9:　$Input_t$ の前処理
10:　$Output_t \Leftarrow$ 学習済み深層学習モデルを $Input_t$ に適用
11:　$Output_t \Leftarrow (Output_t + Output_{t-1})/2$
12:　**if** $Output_t > \sigma$ **then**
13:　　なりすましユーザ検知アラート
14:　**end if**
15: **end if**

撃者が内部情報を抜き取るシェルスクリプトを用意して攻撃を行う場合を考えると，P は可能な限り小さいほうが望ましい．しかし小さすぎるとユーザの通常行動との違いを判定するに十分な情報が揃わない段階で判定することになり，適切な値に設定する必要がある．データ作成時間範囲 T は，検知のために参照する履歴の長さを決めるパラメータである．この値が小さすぎると参照する情報が少なくなり，大きすぎると古い情報を参照するため，前日の入力コマンド情報を参照するデータなど問題が発生するため適切に決定する必要がある (Alg.1,2 行目)．また情報が少なすぎて識別がそもそも困難な場合があるため，分析に用いるデータは mn コマンド以上，深層学習モデルの層の数を考慮して mx 未満としており，mx 未満のコマンドで構成されるデータは mx コマンドになるようにパディング処理をしている (Alg.1,3-8 行目)．この入力データに対し，前処理を行ってから学習済み深層学習モデルを適用し，出力されたコマンドログの異常度が一定の閾値を超えたときアラートを出す流れになっている (Alg.1,9-14 行目)．データの前処理については以下の通りである．

- 入力コマンドの先頭と最後に空白文字があれば消去
- 連続している空白文字を 1 つに統一
- 半角英字を小文字に統一
- 半角英数字記号以外の文字を除去
- ユーザ ID など個人特有の情報をマスキング処理
- 25 文字以上連続している半角英数字を除去

本研究の実証実験では，$P = 1$, $T = 21600$, $mn = 5$, $mx = 20$ でコマンドログを作成し，深層学習モデルの評価を行っている．時間の比較では Unixtime 形式に変換して行っている．データ作成の具体例として図 2 を示す．現状

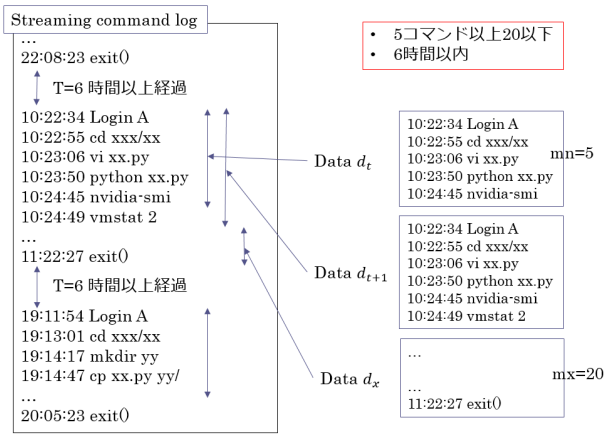

図 2　入力データ作成の具体例

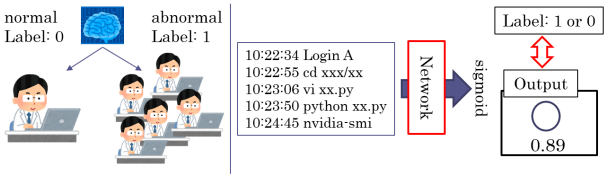

図 3　なりすましユーザ検知モデルの学習

のなりすまし検知フローはコマンドログの異常度判定が閾値を超えたら検知する流れであり，局所的なコマンドの集合で判定を行っているため誤警報率が高くなりやすい．誤警報率を下げるために前回の出力値との移動平均をとっているが，より実用的な誤警報率を実現するためには，コマンドログを包括的に捉えてアラートを出すシステムが望ましい．本研究では，上記の特性を実現するため，自己注意機構とシステム応答を入力情報に導入した深層学習モデルを提案する．

2.2　正規ユーザを識別する深層学習モデル

深層学習を用いたコマンドログに基づくユーザなりすまし検知モデルを提案する．学習の過程は図 3 に示すように，学習は一対他の形式，つまり正規ユーザ (ラベル 0) かそれ以外のユーザ (ラベル 1) を識別するモデルを作成している．

前述したようにコマンドログに基づくなりすましユーザの検知に深層学習を用いた先行研究として，Elmasry らの研究がある．Elmasry らは図 3 の検知モデルのネットワーク部分に，DNN と LSTM[6]，Chracter-level CNN[5] の 3 つの深層学習モデルで実験しており，Chracter-level CNN を用いたネットワークの方が優れていると結論づけている．これは分析対象のデータはコマンドログであり，一般の自然言語処理タスクに出現しない単語や誤字が多く出現しやすいことから，単語よりも文字で特徴を捉える N-gram を利用した方が望ましいためである．

本研究では，Chracter-level CNN をベースとして，入力コマンドの引数と次の入力コマンドを区別して学習できるようにログデータを拡張し，1 コマンド内で畳み込みを行ったあとに自己注意機構や BiLSTM でコマンド間の出現情報を学習するモデルを提案する．従来の Chracter-level CNN モデルでは，図 4 に示すように，入力コマンドを時系列順に最大文字数 M を決めて並べ，埋め込み層で文字それぞれの L 次元の分散表現を取得し，$M \times L$ の 2 次元特徴マップから畳み込みを行う手法である．N-gram の形式で畳み込みを行うフィルタを作成し，局所的な文字の共起情報を特徴抽出する．

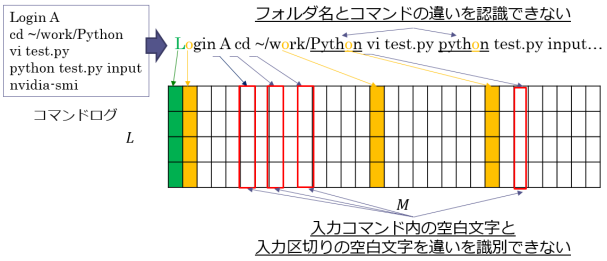

図 4　Character-level CNN を用いた解析の課題点

ここで従来手法の課題点について説明する．1 つは畳み込みを行う際，引数を伴う入力コマンドとの関係をうまく学習できない点である．図 4 の 5 コマンドの具体例にそって説明すると，このコマンドログは (1) ユーザ A がシステムにログインし，(2)cd で作業フォルダに移動して，(3)vim で test.py のスクリプトファイルを編集し，(4)test.py のスクリプトファイルを引数 input を与えつつ実行して，(5) その後の GPU 動作を確認している，といった流れである．このログデータを一次元方向に入力順に並べて特徴抽出すると，隣り合う文字を捉えるフィルタを利用するため，直前に入力したコマンドの引数の一部と次入力するコマンドの共起性を学習することになる．具体例だと，7-gram のフィルタにおいて，フォルダ名の ‘Python’ とテキスト編集コマンド ‘vi’ の隣接を捉えた特徴が生成される．しかし，この場合ユーザの特徴を捉えるために学習したいことは，“cd　/work/Python” の後に ‘vi test.py’ が入力されたという情報と，‘cd’ と ‘ /work/Python’，‘python’ と ‘test.py’ と ‘input’ にそれぞれ共起性が存在するという情報である．

2 つ目の課題点は入力コマンドの文字数の偏りに対応できていない点である．モデルを構成する際に入力として受け取る最大文字数 M を設定する必要があるが，その文字数を超えると情報が切り捨てられるため，コマンドログの情報が全体を考慮せず，局所的な入力になる．極端な例を挙げると，最大文字数を $M = 200$ としたとき，入力された 20 コマンドのログデータが前半の 10 コマンドで 200 文字あった場合，後半の 10 コマンドは切り捨てられて学習してしまう．

これらの課題点に対処するため，入力コマンド数，入力コマンドの文字数，文字ベクトルの次元数の 3 タイプのデータに拡張したモデルを提案する．提案モデルを図 5 に

Vol.2020-DPS-182 No.21
Vol.2020-CSEC-88 No.21
2020/3/12

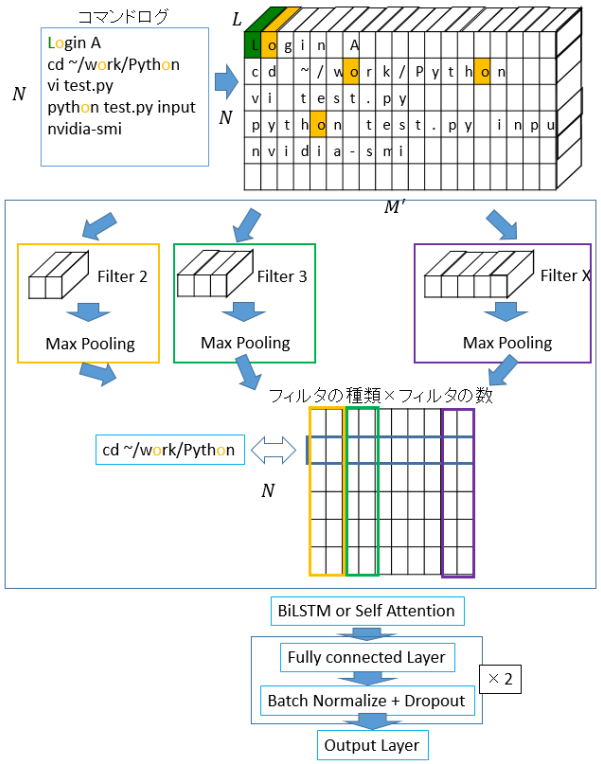

図 5　提案モデルの概要図

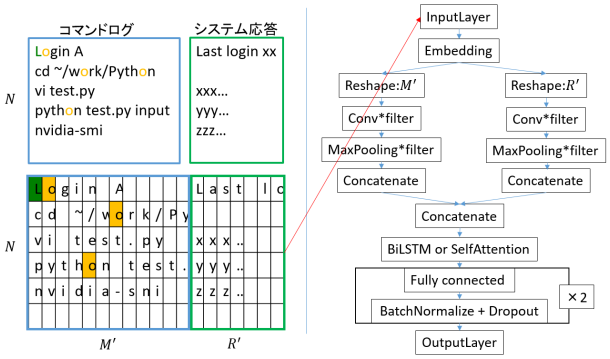

図 6　入力コマンドに対するシステム応答の追加

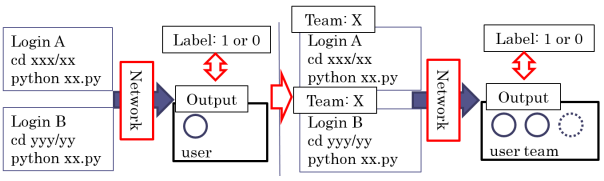

図 7　マルチラベルモデルによる学習

示す．入力データであるコマンドログは，Alg.1 より生成され，普段の入力コマンドと違う異常なものか判定する．

図 5 の N は入力データの最大コマンド数，M' は 1 回のコマンド入力における最大文字数であり，L は文字の分散表現を得るときの次元数である．つまり，$N = mn$ になる．入力は $N \times M'$ 次元のベクトルで，その要素は Unicode の文字 ID に対応している．そのベクトルを埋め込み層により文字ごとの分散表現を取得し，$N \times M' \times L$ のベクトルを得る．それから，1 コマンド入力内の Character-level CNN を N 回行う．つまり，3 文字ずつ畳み込むフィルタで特徴抽出を行う場合，$1 \times 3 \times L$ のフィルタで特徴を抽出する．畳み込みを行うフィルタの種類とその数の積を F としたとき，最大プーリングを通し連結して得られる特徴ベクトルは $N \times F$ である．

この特徴ベクトルは隣接するコマンドの関係を捉えていないため，出現順序を重視して学習する BiLSTM[7] やコマンド間の類似性と共起性を重視する自己注意機構 [8] を導入する．これにより，前述した Character-level CNN の課題点に対処している．その後，複数の全結合層を通してコマンドログの異常度を計算する．

2.3　入力コマンドのシステム応答を含めた学習

前節で提案したモデルの拡張として，入力コマンドに対するシステムの応答出力を含めて学習するモデルについて述べる．これは単純に，図 6 に示すように，前節のネットワークの入力部分にシステムの応答出力のベクトルを追加している．それに伴い，1 コマンドに対するシステム応答の最大文字数を R' とし，入力データのベクトルは $N \times (M' + R')$ で構成される．同様に埋め込み層で文字の分散表現を取得し，$N \times M' \times L$ と，$N \times R' \times L$ に分割してそれぞれに Character-level CNN を適用している．以降の計算過程は，前節と同様である．

コマンドに対するシステム応答を加味するのは，システム応答にもユーザの意図が反映されると考えたためである．そもそもファイルやディレクトリの情報を示す ‘ls’ コマンドやディレクトリを移動する ‘cd’ などのコマンドは，ほとんどのユーザに共通して出現するためコマンド自体にユーザを識別可能な情報は含まれない．しかし，同じ ‘ls’ コマンドでも普段アクセスしない場所と，普段作業する場所での入力では違いが生じる．システム応答も同時に考慮することで，ユーザの識別精度が向上することを検証する．

2.4　マルチラベルモデルによる学習

本研究で提案しているなりすましユーザの検知モデルは，ユーザそれぞれに対して学習を行うものである．しかし，そもそもモデルを学習するのに十分なデータを確保できない場合は珍しくなく，新規ユーザでデータが多く取れない場合，急に所属する部署が変わって入力コマンドの傾向が変わった場合など，入力コマンドが異常であるかどうかの単純な二値判定では誤検知率を下げることは難しい．そこで，入力コマンドがユーザの所属しているチームでよく出現するコマンドであるのか同時に判定するマルチラベルモデルにすることで，データの少なさやユーザの所属が変化したときにある程度考慮可能な学習にしている．図 7 に示す．

図 3 に示すように，今回のなりすましユーザ検知モデル

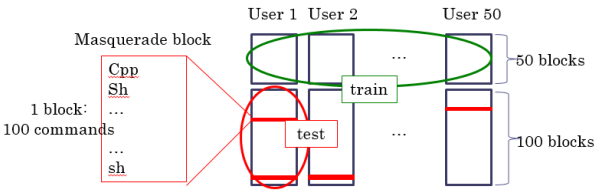

図 8　SEA dataset の構成

の学習は対象のユーザとそれ以外のユーザ，つまり一対他の形式で学習を行っている．しかし，これは実際になりすましユーザのコマンドが含まれているわけではない．なりすましによる不正アクセスの目的は顧客データ等の不正入手を目的にしたものが多いこと [1] を考慮すると．将来的には内部情報を探るときに用いる攻撃者コマンドが含まれているか同時に判定するラベルを追加するなどして，より実践的な検知モデルにすることが重要であると考えられる．

3. なりすましユーザ検知モデルの性能評価

3.1 公開データセットを用いた提案モデルの性能評価

Schonlau ら [9] が作成した UNIX コマンドラインベースのなりすまし検知のデータセット (以下 SEA dataset) を利用して評価した．

3.1.1 SEA Dataset と実験設定

Alg.1 の流れで統一し，深層学習モデルによる異常度判定の部分で提案モデルと従来手法の Character-level CNN を適用することで性能評価を行った．SEA dataset の構成を図 8 を示す．

SEA dataset は Linux のセキュリティ監査ツールを利用し，70 人のユーザに対して数ヶ月以上にわたって収集されて作成されている．70 人中 50 人のコマンドログデータを検知対象とし，残りの 20 人を攻撃者データとして利用している．50 人のユーザそれぞれに対して 15000 コマンドが与えられていて，100 コマンドで 1 ブロックに分割可能であり，ユーザそれぞれ 150 ブロックで構成されている．最初の 50 ブロック (5000 コマンド) は対象のユーザであることが保証されていて，残りの 100 ブロック (10000 コマンド) には 20 人の攻撃者データからランダムに選択されたブロックが混入しており，この混入されたブロックを検知することが課題である．つまり後半の 100 ブロック (10000 コマンド) はテストデータに利用されて，トレーニングデータでは前半の 50 ブロック (5000 コマンド) を正規ユーザラベル，他の 49 人の 50 ブロックからなる 2450 ブロックがなりすまし攻撃者ラベルとして利用可能である．

データは入力コマンドの先頭のみで構成されているため，複数の引数は含まれていない．また時系列順にコマンドは並べられているが詳細な時間は分からない．そのため，入力データとなるコマンドログの作成は，Alg.1 の T は考慮せず，$mx = 20$，$mn = 5$，$P = 1$ で作成した．

1 ブロック (100 コマンド) ごとに適応し，5 コマンド以上 20 コマンド以下になるように入力データを作るため，1 ブロックあたり 95 のコマンドログが生成される．識別モデルはユーザごとに作られるため，学習に用いるデータは対象ユーザの 50 ブロックとそれ以外のユーザの 2450 ブロックである．そのまま利用すると学習データに大きな偏りができるため，ラベルの比が一対一にダウンサンプリングして学習データを作成する．テストデータは 100 ブロックにそのまま適用して作成している．つまり，9500 のコマンドログが生成されてユーザのなりすましを評価した．

また，このデータセットの特徴としてテストに用いる 100 ブロックになりすましのブロックが混入していない場合，つまり 100 ブロック全て正規ユーザの場合が存在する．そのため，深層学習モデルの評価指標として RMSE も採用した．この評価指標は全データに対する回帰の誤差を表している．また，なりすまし攻撃者のブロックが混入している場合のユーザは検知率を測定できるため，そのモデルの性能評価として，AUC と閾値を適切にとった場合の F1-score を測定した．

以上の実験設定をもとに，比較手法として以下の 5 つのパターンで実験した．提案手法の Character-level CNN は，従来手法と区別して PR Character-level CNN と表記する．それぞれのハイパーパラメータは以下の通りである．

- PR Character-level CNN + BiLSTM
- PR Character-level CNN + SelfAttention
- PR Character-level CNN
- Character-level CNN
- BiLSTM + SelfAttention

CNN ベースの 4 つの手法は，文字ベクトルの次元数は 128，N-gram のフィルタ形式で (2，3，4) の 3 種類かつフィルタの数は 128，活性化関数は最後の出力層はシグモイドでそれ以外は relu 関数を用いた．全結合層のユニット数は 256，ドロップアウト率は 0.4 である．バッチサイズは 64，学習には Adam を使用した．Character-level CNN の入力データの最大文字数は M を 300，PR Character-level CNN の 1 コマンドの最大文字数は M' を 10 で統一した．BiLSTM + SelfAttention の手法は単語ベースと文字ベースの手法比較の一例として実験している．

3.1.2 コマンドログによるユーザ識別の性能評価

SEA dataset を用いた評価実験の結果を表 1 に示す．ユーザが 50 人に対して検知モデルを作成し，その評価指標の平均値で比較を行っている．しかし，SEA dataset の特徴として，テストデータの 100 ブロックになりすましユーザのブロックが混入していないパターンが 21 あるため，AUC と F1-score はなりすましユーザのブロックが混入しているユーザで平均値を計算している．つまり，RMSE の計算はユーザ 50 人の平均値，AUC と F1-score はユーザ 29 人の平均値であることに注意されたい．

RMSE は出力値とラベルの誤差に関する評価指標であ

表 1　深層学習モデルの性能評価

深層学習モデル	RMSE コマンドログ	RMSE ブロック	AUC コマンドログ	AUC ブロック	F1-score コマンドログ	F1-score ブロック
PR Character-level CNN + BiLSTM	0.3919	0.3373	0.8855	0.9142	0.5518	0.7612
PR Character-level CNN + SelfAttention	0.4468	0.3931	0.8925	0.9311	0.5827	0.8000
PR Character-level CNN	0.4174	0.3643	0.8798	0.9132	0.5496	0.7541
Character-level CNN	0.4227	0.3574	0.8882	0.9152	0.5639	0.7387
BiLSTM + SelfAttention	0.4577	0.3887	0.8825	0.9111	0.5009	0.7333

るため，値が小さいほど回帰モデルとして優れている．RMSE の評価指標で深層学習モデルを比べた結果，PR Character-level CNN と BiLSTM を組み合わせたモデルが最もよかった．しかし，入力データが異常か正常か二値分類するにあたって，AUC と F1-score の評価指標で比較すると，PR Character-level CNN と自己注意機構を組み合わせたモデルが最も良いことが分かった．PR Character-level CNN に BiLSTM と自己注意機構を組み合わせる目的は，入力コマンド間の関係を捉えるためである．BiLSTM や自己注意機構がないモデルでは性能が悪くなっているため，入力コマンド間の情報は有用であることが分かった．従来手法の Character-level CNN は，入力データの文字を横並べして畳み込みを行うため，隣接した文字同士の関係を捉えた特徴を抽出している．提案モデルと従来手法は特徴抽出の意味合いでは同じであるため，性能に大きな違いが生じない結果となった．これは，SEA dataset が入力コマンドの先頭のみを抽出してデータが作られたことにも起因していると考えられる．また，Lin ら [10] の研究を参考に，文字単位ではなく単語単位でベクトルを取得し，BiLSTM と自己注意機構で学習するモデルを実装し評価したが，Character-level CNN を利用した手法の方が優れていることが確認できた．

3.2　実証データへの適用とその性能評価

LINE 株式会社より提供いただいた実証データを用いて性能評価を行った．

3.2.1　LINE Dataset と実験設定

実証データは会社のシステム運用に携わるチームのユーザ 33 人を対象に，なりすましユーザを検知するモデルを作成することを試みた．学習に用いるデータは 2019 年 4 月 1 日から 2019 年 11 月 30 日までの 8 ヶ月間であり，この学習済みモデルを 2019 年 12 月 1 日から 12 月 31 日までの 1 ヶ月分のテストデータに適用して評価した．データは入力コマンドとその出力結果，タイムスタンプが含まれており，モデルの入力データとなるコマンドログの作成は，Alg.1 に基づいている．$mx = 20$，$mn = 5$，$P = 1$ に加えて，タイムスタンプが利用できることから $T = 21600(s)$ で設定した．ユーザによってデータ数が大きく異なるため，SEA dataset のようにブロック単位の分析は行っていない．またコマンドログのサイズが大きいユーザは，直近の 10000 コマンドログを利用した．

表 2　実証データにおける深層学習モデルの性能評価

深層学習モデル		RMSE コマンドログ	AUC コマンドログ	F1-score コマンドログ
入力データにシステム応答あり	PR Character-level CNN + BiLSTM	0.1692	0.9782	0.9715
	PR Character-level CNN + SelfAttention	0.1481	0.9868	0.9788
入力データにシステム応答なし	PR Character-level CNN + BiLSTM	0.2249	0.9096	0.9127
	PR Character-level CNN + SelfAttention	0.2320	0.9275	0.9376
	Character-level CNN	0.2127	0.9071	0.9192

学習データは第 2.4 節より，対象ユーザのコマンドログと対象ユーザが所属するチームユーザのコマンドログ，全く関係ないチームユーザのコマンドログより構成される．本実験では，対象のユーザのコマンドログの上限を 6000 としてランダムに抽出し，チームユーザのコマンドログ総数の上限 1500 になるように均等にユーザからコマンドログを取得し，この二つのデータの和と比率が同じになるように，全く関係のないチームユーザからそれぞれコマンドログを取得して学習データを作成している．テストデータに関しても同様の処理でデータを作成している．

以上のデータセットに対し，次の 3 つの手法で精度比較を行った．

- PR Character-level CNN + BiLSTM
- PR Character-level CNN + SelfAttention
- Character-level CNN

また，入力コマンドに対するシステム応答を入力データとして利用可能であるため，第 2.3 節で提案した学習モデルを検証した．ハイパーパラメータは第 3.1 節の実験と同様に設定している．ただしシステム応答を使う場合，R' を 40 に設定して実験した．

3.2.2　コマンドログによるユーザ識別の性能評価

評価実験の結果を表 2 に示す．この実験で用いた実証データは入力コマンドとその引数も考慮して学習している．

SEA dataset は入力コマンドの先頭のみを用いて構成されていたため，その違いによる精度の変化を確かめた．表の入力データにシステム応答を考慮しなかった場合の結果より，PR Character-level CNN に自己注意機構を組み合わせたモデルが，分類モデルの指標 AUC と F1-score において最も良い結果となった．この実験結果は，SEA dataset の実験結果の表 1 と傾向が似ており，異常なユーザと正規ユーザを分類する点においては，自己注意機構を用いた学習過程の方が望ましいことが推測される．また，自己注意機構は計算コストの面でも優れているため，なりすましユーザ検知のタスクを自動化する点においても望ましい．提案手法は，従来手法と比較して入力コマンドの引数との関係を区別して学習できるように意図して構築されたが，今回の実験結果では約 2 %程しか精度の向上がみられなかった．これにより，1 つ前の入力コマンドの引数とその後の入力コマンドの先頭部分を捉えた学習は，学習にそれほど大きな悪影響を及ぼさない可能性も考えられる．

3.2.3 システム応答を含めたモデルによる性能評価

入力コマンドに対するシステム応答の文字列情報を追加することで精度の向上を期待したモデルであり，実証データに適用して精度の変化を確認した．表 2 よりシステム応答を考慮することで AUC が 6 %以上良くなっていることがわかる．本実験の入力データであるコマンドログは 5 以上 20 以下のコマンドで構成されている．これは早期検知に重きをおいて設定しているが，‘ls’ など多くのユーザが使うコマンドしか含まれていない入力データも検知対象になっているため誤検知が多い．そこにシステム応答の情報が加わり識別可能になったデータが多く存在していることが確認された．

4. まとめ

本論文では，UNIX 系サーバーで収集された社員の実行コマンド・ログ情報に基づいて，なりすましによる不正アクセスユーザを判定する深層学習モデルを提案し，2 つのデータセットで性能を検証した．具体的には，1 つのコマンド入力で完結した Character-level CNN で特徴ベクトルに変換し，BiLSTM や自己注意機構を導入して，時系列情報や重要な共起性に焦点を当ててコマンドログの異常を判定するモデルを提案した．これは従来手法の Character-level CNN を用いたアプローチが，入力データを単純に横並べして学習を行っており，入力コマンドの引数などが混在した際にコマンドの時系列情報が曖昧になる点を改善しようと試みたモデルである．

従来手法との性能比較の結果，提案した Character-level CNN と自己注意機構を組み合わせたモデルが，両方のデータセットで最も高い検知精度を示した．また，実行コマンドに対するシステム応答の文字列を同時に考慮して学習することで，なりすましユーザの検知精度が向上することを示した．

しかし，なりすましユーザ検知システムとしての実用性の検証はまだ十分でない．一対他の形式でユーザそれぞれにモデルを作成しているため，実践的な危険なコマンドの傾向を捉えて学習できていない問題点がある．また，なりすましユーザの検知においてコマンド情報をどのぐらい考慮してデータを作成し，どの基準を超えたらアラートを出すのが最も良いのか調査できていない点などが課題として挙げられる．

謝辞 本研究を進めるにあたり, データセットの提供から，研究への助言等サポートしてくださった LINE 株式会社の方々に深く感謝の意を表します.

参考文献

[1] 総務省, 不正アクセス行為の発生状況及びアクセス制御機能に関する技術の研究開発の状況, 2019.
[2] Kim, Han-Sung and Cha, Sung-Deok, “Empirical Evaluation of SVM-Based Masquerade Detection Using UNIX Commands,” *Computers & Security*, pp. 160–168, Elsevir, 2005.
[3] Amruta, Mahajan, “Masquerade Detection Based on UNIX Commands,” *San Jose State University*, 2012.
[4] Elmasry, Wisam, Akbulut, Akhan and Zaim, Abdul Halim, “Deep Learning Approaches for Predictive Masquerade Detection”, *Security and Communication Networks* , Hindawi, 2018.
[5] Zhang, Xiang, Zhao, Junbo, and LeCun, Yann, “Character-Level Convolutional Networks for Text Classification,” *Advances in Neural Information Processing Systems*, pp. 649–657, 2015.
[6] Hochreiter, Sepp and Schmidhuber, Jürgen, “Long Short-Term Memory,” *Neural computation*, Vol. 9, No. 8, pp. 1735–1780, MIT Press, 1997.
[7] Graves, Alex and Schmidhuber, Jürgen, “Framewise phoneme classification with bidirectional LSTM and other neural network architectures,” *Neural Networks*, Elsevier, 2005.
[8] Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan, N and Kaiser, Łukasz, and Polosukhin, Illia, “Attention is All You Need,” *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.
[9] Schonlau, Matthias, DuMouchel, William, Ju, Wen-Hua, Karr, Alan F, Theusan, Martin, Vardi, Yehuda and others, “Computer Intrusion: Detecting Masquerades,” *Statistical Science*, Vol. 16, pp. 58–74, Institute of Mathematical Statistics, 2001.
[10] Lin, Zhouhan, Feng, Minwei, Santos, Cicero Nogueira dos, Yu, Mo, Xiang, Bing, Zhou, Bowen, and Bengio, Yoshua, “A Structured Self-Attentive Sentence Embedding,” arXiv:1703.03130, 2017.
[11] Okamoto, Takeshi, Watanabe, Takayuki, and Ishida, Yoshiteru, “Towards an Immunity-Based System for Detecting Masqueraders,” *Int. Conf. on Knowledge-Based and Intelligent Information and Engineering Systems*, pp. 488–495, Springer, 2003.